프로젝트

참고

퍼징 테스트 기반 취약점 도구

https://github.com/wapiti-scanner/wapiti

https://github.com/NESCAU-UFLA/FuzzingTool

취약점 분석 도구

https://github.com/neuroo/grabber

https://github.com/v3n0m-Scanner/V3n0M-Scanner

https://github.com/volatilityfoundation/volatility

분야	점검항목	세부 점검항목	점검항목 설명
중요정보주) 보호	메모리 내 노출 방지	메모리 내 중요정보 평문 노출 확인	□ 단순 로그인* 외 중요정보 입력 또는 처리 과정 등**에서 메모리 내 평문(또는 쉽게 평문으로 변환 가능한 문자열) 노출 여부를 점검 * 웹 애플리케이션 최초 메뉴 진입을 위한 로그인(전자금융거래의 직접적인 처리와 무관) ** 예) 주민등록번호 입력, 간편결제 비밀번호 입력 등
	웹 영역 내 노출 방지	웹 영역 내 중요정보 평문 노출 확인	□ 중요정보의 입력 또는 처리 과정 등에서 HTML, Javascript, DOM 등 웹을 표현하기 위한 영역 내 평문(또는 쉽게 평문으로 변환 가능한 문자열) 노출 여부를 점검
	네트워크 구간 내 노출 방지	중요정보 평문 전송 확인	□ 중요정보의 네트워크 구간 내 평문(또는 쉽게 평문으로 변환 가능한 문자열) 노출 여부를 점검
		취약한 HTTPS 정책 사용 확인	□ 취약한 HTTPS 정책 사용 여부를 점검 - 서버 인증서의 유효기간 만료 여부 확인 - 인증서의 서명 알고리즘 및 서명 해시 알고리즘에 SHA1 이하의 해시 알고리즘 사용 여부 확인 - 취약하거나 보안수준이 낮은 프로토콜(SSL2.0/3.0, TLS1.0/1.1) 사용 여부 확인 - 보안 강도가 낮은 암호 알고리즘 허용 여부 확인 - 취약한 방식의 HTTPS 재협상 허용 여부 확인 - 취약점이 존재하는 HTTPS 확장 모듈의 존재 유무 확인 - 전방향 안전성(Forward Secrecy) 지원 여부 확인 ※ 취약한 HTTPS 정책 확인 및 조치 시 HTTPS 정책 검증 도구(https://ssllabs.com 등) 참고
	파일 내 노출 방지	파일 내 중요정보 평문 저장 확인	□ 이용자 PC에 저장되는 핀테크서비스 관련 파일 내 중요정보의 평문 저장 여부를 점검

주) 오픈뱅킹 관련 중요정보(오픈뱅킹 인증키(client_secret), 오픈뱅킹 접근키(access_token) 등), 이용자 중요정보(고유식별정보, 비밀번호(로그인 비밀번호, 거래 비밀번호 등), 기타 중요 인증정보 등) 등

분야	점검항목	세부 점검항목	점검항목 설명
	화면 내 노출 방지	화면 내 중요정보 평문 노출 확인	□ 중요정보의 화면 내 평문(원본*) 노출 여부 및 화면캡처를 통한 탈취 가능 여부를 점검 * 본인인증을 위해 신분증 이미지파일을 업로드하는 경우, 신분증에 기재된 중요정보가 마스킹되지 않은 원본 이미지파일을 의미
	입력정보 보호 적용	입력정보 보호대책 적용 확인	□ 이용자의 중요정보 입력 시 입력정보의 평문(또는 쉽게 평문으로 변환 가능한 문자열) 노출 방지를 위한 보호기능 적용 여부를 점검
거래정보 위·변조	계좌정보 변조 방지	계좌정보 무결성 검증 확인	□ 전자금융거래 이용 중 계좌정보 위·변조 시 타인계좌 조회 및 위·변조된 계좌로 거래 가능 여부를 점검
	금액정보 변조 방지	금액정보 무결성 검증 확인	□ 전자금융거래 이용 중 금액정보 위·변조 시 위·변조된 금액으로 거래 가능 여부를 점검
		금액 한도 검증 확인	□ 전자금융거래 이용 중 이체한도 범위를 벗어나는 금액(최대한도 초과, 음수 등)으로 거래 가능 여부를 점검
	거래정보 재사용 방지	거래정보 재사용 확인	□ 전자금융거래 진행 시 이용된 거래정보를 재전송하여 재사용 가능 여부를 점검
서버 보안	서버 보안 적용	리다이렉트 기능을 이용한 피싱 가능성 확인	□ 리다이렉트 기능이 존재하는 경우 URL 인자값을 임의의 페이지로 변경하여 이동 가능 여부를 점검
		불필요한 메소드 허용 확인	□ 불필요한 HTTP 메소드 허용 여부를 점검
		외부 사이트에 의한 운영정보 노출 확인	□ 검색엔진 등을 통해 핀테크서비스 관련 중요정보의 획득 가능 여부를 점검
		SQL 인젝션 확인	□ 이용자의 제어가 가능한 파라미터에 의해 SQL 쿼리문이 완성되는 점을 이용하여, 해당 파라미터 변조를 통해 SQL 쿼리문 조작 가능 여부를 점검
		디렉토리 목록화 확인	□ 특정 디렉토리에 초기 페이지가 존재하지 않거나, 웹서버의 디렉토리 인덱싱 허용 설정 등으로 인한 해당 디렉토리 내 하위 디렉토리 및 파일 목록 노출 여부를 점검

분야	점검항목	세부 점검항목	점검항목 설명
		서버 운영정보 노출 확인	□ 응답헤더 및 에러메시지 등에서 서버 운영정보(버전, 절대경로 등) 노출 여부를 점검
		크로스사이트 스크립팅(XSS) 확인	□ 스크립트 전송 또는 업로드를 통해 응답값에 스크립트 포함 및 실행 가능 여부를 점검
		크로스사이트 리퀘스트 변조(CSRF) 확인	□ 스크립트 구문 등을 업로드하여 타 이용자의 권한으로 실행 가능 여부를 점검
		파일 업로드 취약점 확인	□ 파일 업로드 기능을 이용하여 서버 사이드 스크립트 파일(JSP, ASP 등) 업로드 및 실행 가능 여부를 점검
		파일 다운로드 취약점 확인	□ 파일 다운로드 기능을 이용하여 접근이 허용된 디렉토리 외의 경로에 위치한 중요파일(소스코드, 설정파일 등) 다운로드 가능 여부를 점검
		관리자 페이지 노출 확인	□ 유추하기 쉬운 관리자 페이지(/admin, /manager 등) 경로에 접근 가능 여부를 점검
		불필요 파일 노출 확인	□ 잘 알려진 경로 대입 등을 통해 불필요 파일(테스트 파일, 백업 파일 등) 노출 여부를 점검
		고정된 state 변수 부여 확인	□ OAuth 이용자 인증 요청 시 가변적이고 추측이 어려운 state 변수값 부여 여부를 점검
		state 변수 무결성 검증 확인	□ 인증 완료 요청시 state 변수값을 변조할 경우 state 변수에 대한 무결성 검증 여부를 점검
		(권고) 안전한 인증페이지 제공 확인	□ OAuth 이용자 인증 과정에서 iframe을 이용하여 외부 도메인을 호출하는 경우 이용자의 인증페이지 식별*이 제한되어 피싱 위험 등이 존재하므로 안전한 방식**을 통한 인증페이지 출력을 권고 * 도메인 주소의 철자, 자물쇠 그림 확인 등 ** 인증페이지 팝업 등
인증	인증 우회 방지 적용	이용자 인증정보 재사용 확인	□ 이용자 인증 요청(로그인 요청 등) 시 사용된 인증정보를 획득 후 재전송하여 이용자 권한 획득 가능 여부를 점검

분야	점검항목	세부 점검항목	점검항목 설명
		세션정보 재사용 확인	□ 이용자 세션정보를 획득하여 별도의 환경에 강제 적용 후 해당 이용자 권한 획득 가능 여부를 점검
		불충분한 세션 만료 확인	□ 이용자 인증 후 서비스 이용 없이 일정시간 경과 후 또는 이용자 로그아웃 요청 후 세션 종료 여부를 점검
		비밀번호 복잡도 검증 수준 확인	□ 전자금융거래, 로그인 등의 절차 진행 시 요구되는 비밀번호의 복잡도 검증 수준을 점검
		비밀번호 오류 횟수 제한 확인	□ 전자금융거래, 로그인 등의 절차 진행 시 이용자가 입력하는 비밀번호의 오류 횟수(5회) 제한 여부를 점검
		불충분한 이용자 인증 확인	□ 이용자 인증이 요구되는 페이지*에 접근 시 추가 인증 요구 여부 및 URL 직접 호출, 플로우 통제 우회 등을 통한 인증 우회 가능 여부를 점검 * 비밀번호 변경 페이지, 민감정보가 포함된 페이지 등
		부적절한 비밀번호 초기화 확인	□ 비밀번호 복구 절차 이용 시 타 이용자의 비밀번호 획득, 변경 등의 가능 여부를 점검
		부적절한 인증정보 발급 확인	□ 서버가 발급하는 인증정보를 추측하여 타 이용자의 권한을 획득할 수 없도록 인증정보의 적절성(가변성 및 복잡성 등)을 점검
		불충분한 인가 확인	□ 파라미터 변조 등을 통한 타 이용자의 민감정보 노출 또는 권한 탈취가 이루어지지 않도록 접근 권한 검증 여부를 점검
		쿠키정보 변조 확인	□ 쿠키정보를 이용한 이용자 검증 등의 절차가 존재하는 경우, 쿠키정보 변조를 통한 권한 탈취 등의 취약점 존재 여부를 점검
		이체성 거래 시 인증 적용 확인	□ 이체성 거래 시 인증(예: 거래 비밀번호 확인) 적용 여부를 점검
		(권고) 계좌 소유주 검증 확인	□ 계좌 소유주 인증 시 계정 소유주와 일치 여부를 확인하지 않을 경우 검증 절차 적용을 권고
		(권고) 소셜 로그인 이용 시 계정 보안 강화 확인	□ 소셜 로그인(Google 로그인 등)을 이용하여 이용자를 관리하는 경우 소셜 로그인 과정에서 강화된 인증절차 적용 안내를 권고

정보 누출

보안위협: 웹 사이트에 중요정보(개인정보, 계정정보, 금융정보 등)가 노출되거나 에러 발생 시 과도한 정보(애플리케이션 정보, DB 정보, 웹 서버 구성 정보, 개발 과정의 코멘트 등)가 노출될 경우 공격자들의 2차 공격을 위한 정보로 활용될 수 있음

점검방법:

1) 웹 사이트에 중요정보가 평문으로 노출되고 있는지 확인

2) 웹페이지에 마스킹 된 중요정보가 웹페이지 소스에 평문으로 노출되고 있는지 확인

3) 에러 메시지 또는 에러 페이지에서 과도한 정보가 노출되는지 확인

4) 인코딩된 중요정보는 디코딩 가능한지 확인

5) 임의의 계정으로 로그인을 시도하여 반환되는 에러 메시지를 통해 특정 ID의 가입 여부를 식별할 수 있는지 확인

```
코드 내 중요 정보 노출(평문 전송) 취약점중요한 정보가 암호화되지 않은 상태로 전송되어서 스니핑과 같은 공격을 
 통해 탈취되는 취약점.중요정보평문전송민감한 정보가 암호화되지 않고 평문으로 전송될 때 발생하는 취약점.
이 취약점을 확인하는 방법 2가지:중요한 정보를 전송하는 페이지가 HTTPS(SSL)와 같이 암호화된 채널로 통신하는
지 확인소스코드 내에서 중요한 정보를 암호화 함수를 통해 전송하는지 확인
암호화 할떄에는 쉽게 복호화 되지 않는 방법으로 암호화하는것이 중요하다.
하드코딩된 패스워드 하드코딩: 패스워드와 같은 민감한 정보가 소스코드에 그대로 노출되어 있는 경우.주석처리된중
요정보주석에 중요 정보가 포함되어 있는 경우.
```

메모리 내 노출 방지

메모리 내 중요정보 평문 노출 확인

→ 평문 암호화 적용여부 확인, https 통신 여부 확인

선규님이 올려준 자료의 httpCheck(url) 함수는 단순 url에 https 를 확인하는 방식 → SSL/TLS 암호화 인지 확인하는 방법이 필요함.

OWASP TOP 10

1. Broken Access Control (취약한 접근 제어)

접근 제어는 사용자가 권한을 벗어난 행동을 할 수 없도록 정책을 만들고 실행하는 기능입니다. 접근 제어가 취약하게 구현되면 사용자는 주어진 권한을 벗어나 인가되지 않은 데이터에 무단으로 접근해, 조작이나 삭제하는 등을 할 수 있습니다.

관리자 페이지 노출

- 점검 내용: 유추하기 쉬운 URL로 관리자 페이지 및 메뉴 접근의 가능 여부 점검
- 점검 목적: 관리자 페이지 URL이 유추하기 쉬운 이름(admin,manager 등)이나 설정 프로그램 설계 오류로 수정하여 비 인가자의 관리자 메뉴 접근을 방지하고자 함
- 보안 위협: 웹 관리자의 권한이 노출될 경우 홈페이지의 변조뿐 만 아니라 취약성 정도에 따라서 웹 서버의 권한까지도 노출될 수 있음(위험도: 상)
- 대상: 소스코드, 웹 서버, 웹 방화벽

- 판단 기준: 유추하기 쉬운 URL로 관리자 페이지 접근이 불가능한 경우(양호)
- 판단 기준: 유추하기 쉬운 URL로 관리자 페이지 접근이 가능한 경우(취약)

1. 추측하기 쉬운(/admin, /manager, /master, /system, /administrator 등)의 명칭을 사용하는 디렉터리 파일 및 관리자 페이지 존재 여부 확인
2. 추측하기 쉬운(7001, 8080, 8443, 8888 등) 포트의 접속으로 관리자 페이지가 노출되는지 확인
3. 관리자 페이지의 로그인 창에 기본 관리자 계정(admin, administrator, manager) 및 패스워드를 입력하여 로그인되는지 확인
4. 사용자 인증 후 접근한 관리자 페이지 메인 페이지나 하위 메뉴 페이지 등 중간 페이지(/admin/main.jsp, /admin/menu.html 등) URL으로 인증 과정 없이 직접 접근 가능한지 확인

2. Cryptographic Failures

기존 민감 데이터 노출(Sensitive Data Exposure)에서 명칭이 변경되었음.

암호화에 오류가 있거나 미흡한 부분이 있는 경우, 민감 데이터 노출로 이어집니다. 특히 개인정보와 금융 데이터 같은 법과 규정에 강력하게 영향을 받는 경우라면 안전하게 보호하기 위한 추가 요구 사항을 지켜야 합니다.

base64 등의 취약한 암호 알고리즘을 사용했는지 여부를 확인한다.

3. Injection

역사적으로 웹 애플리케이션 취약점 중 가장 많이 알려진 인젝션(Injection)이 세 번째 항목으로 선정됐습니다. 이번에 XSS(Cross-site Scripting)가 인젝션 항목에 포함되면서, 사용자 제공 데이터 조작을 통한 공격은 모두 인젝션 항목으로 통일됐습니다.

인젝션은 사용자가 전달하는 데이터(파라미터, 헤더, URL, 쿠키(Cookie), Json 데이터, SOAP, XML 등 모든 형태)를 신뢰할 수 없는 데이터로 조작해서, 서버 측에서 명령어나 쿼리문의 일부로 인식하게 만들 때 발생하는 취약점입니다.

XSS(Cross Site Scripting)

- 웹 사이트 관리자가 아닌 이가 웹 페이지에 악성 스크립트를 삽입할 수 있는 취약점. 주로 여러 사용자가 보게 되는 전자 게시판에 악성 스크립트가 담긴 글을 올리는 형태로 이루어진다.
- 이 취약점은 웹 애플리케이션이 사용자로부터 입력 받은 값을 제대로 검사하지 않고 사용할 경우 나타난다.

- 이 취약점으로 해커가 사용자의 정보(쿠키, 세션 등)를 탈취하거나, 자동으로 비정상적인 기능을 수행하게 할 수 있다. 주로 다른 웹사이트와 정보를 교환하는 식으로 작동하므로 사이트 간 스크립팅이라고 한다.

Injection

- SQL Injection: 쿼리 등에 SQL 쿼리를 삽입하여 데이터베이스 조회, 변조 및 삭제 등의 공격을 의미한다.
- CRLF Injection: CRLF(%0d%0a) 패턴 삽입을 통한 악의적인 공격을 의미한다.
- Command Injection: 쿼리 등에 운영체제 명령어를 삽입하여, 명령어 실행 및 시스템 권한 획득 공격을 의미한다.

SQL Injection

- Error based SQL Injection: GET, POST 요청필드, HTTP 헤더값, 쿠키값 등에 특수문자(’, “, ;) 삽입 시, SQL 관련 에러를 통해 데이터베이스 정보를 예상해 볼 수 있다.
- Union based SQL Injection: Union은 2개 이상의 쿼리를 요청하여 결과를 얻은 SQL 연산자이며, 공격자는 이를 악용하여 원래의 요청에 추가 쿼리를 삽입하여 정보를 얻어내는 방식이다. 단 Union 쿼리는 2개의 테이블이 동일한 필드 개수와 데이터 타입을 가져야 하므로 사전공격을 통해 해당 정보를 얻는 과정을 거치게 된다.
- Blind based SQL Injection: 에러가 발생되지 않는 사이트에서는 위의 기법들을 사용할 수 없기 때문에 공격을 통해 정상적인 쿼리 여부를 가지고 취약점 여부를 판단하는 기법이다.
- Stored Procedure based SQL Injection: 저장 프로시저는 운영상 편의를 위해 만들어 둔 SQL 집합이나, 명령어 실행이 가능한 MsSQL의 xp_cmdshell을 악용하여 운영체제 명령어를 삽입하는 기법이다.
- Time based SQL Injection: 쿼리 결과를 특정시간만큼 지연시키는 것을 이용하는 기법으로, Blind 기법과 마찬가지로 에러가 발생되지 않는 조건에서 사용할 수 있다.

CRLF Injection

- CRLF는 Carriage Return과 Line Feed를 의미하며, 키보드의 엔터키와 동일한 기능을 한다. 그러나 URL 특정 파라미터에 해당 코드를 삽입하는 경우 임의의 헤더 정보를 생성할 수 있는 취약점이 발생된다.

Command Injection

- GET, POST 요청필드, HTTP 헤더값, 쿠키값 등에 운영체제 명령어를 삽입하여 권한을 획득하는 공격을 의미한다.

4. Insecure Design

안전하지 않은 설계(Insecure Design)는 새롭게 추가된 항목입니다. 코드 구현 단계에 앞서 기획과 설계 단계에서 발생하는 보안 결함을 의미합니다. 안전하지 않게 설계한 애플리케이션은 개발을 완료한 후에 코드를 수정해도 보안 결함을 완벽하게 방어하는데 한계가 있을 수 밖에 없습니다. 그래서 설계 단계에서부터 보안성을 고려해야 합니다.

5. Security Misconfiguration

보안 설정 오류(Security Misconfiguration)는 애플리케이션을 최초 설치하거나 업데이트할 때 보안성을 고려하지 않은 설정으로 인해 취약점이 발생하는 경우로, 설정과 관련된 모든 부분을 포함하고 있습니다. XXE(XML External Entity) 항목은 애플리케이션의 잘못된 보안 설정으로 인해 발생하는 취약점으로, 이번 개정으로 보안 설정 오류 항목에 병합됐습니다.

6. Vulnerable and Outdated Components

취약하고 지원이 종료된 구성 요소(Vulnerable and Outdated Components)는 취약한 버전 또는 EOS/EOL/EOD(소프트웨어 기술 지원 중단) 상태인 소프트웨어를 계속 사용하는 경우를 뜻하며, 그로 인해 발생할 수 있는 모든 보안 위협을 포함하고 있습니다. 서비스를 구성하는 모든 요소(OS, WEB/WAS, 데이터베이스, 애플리케이션, API, 라이브러리, 프레임워크 등)가 여기에 해당됩니다.

7. Identification and Authentication Failures

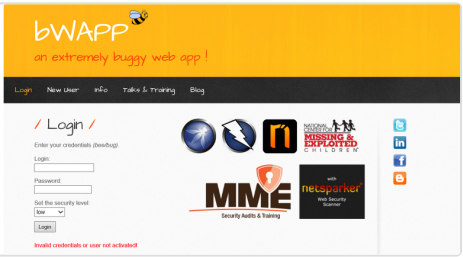
2021년 개정으로 기존 취약한 인증(Broken Authentication) 항목에 식별 실패(Identification failures)를 포함시켜, 조금 더 넓은 의미인 식별 및 인증 실패(Identification and Authentication Failures) 항목으로 변경됐습니다. 사용자 신원 확인과 인증 및 세션 관리에 해당하는 항목으로 2017년 대비 5단계가 낮아져 7번째 항목으로 선정됐습니다만, 여전히 많이 발견되는 취약점입니다.

brute force: 무차별 대입 공격은 특정한 암호를 풀기 위해 가능한 모든 값을 대입하는 것을 의미한다.

Python- 무차별 대입 공격(BruteForce Attack).

무차별 대입 공격: 모든 경우의 수를 무작위로 대입 하여 암호를 푸는 공격 기법 ★ 무차별 대입 공격이 이루어지는 환경은 로그인 기능에 임계값 설정이 없을때 이 공격 기법을 사용한다. 그 이유는 임계값 설

https://webstone.tistory.com/74



8. Software and Data Integrity Failures

소프트웨어 및 데이터 무결성 오류(Software and Data Integrity Failures)는 이번에 신설된 항목으로, 안전하지 않은 역직렬화(Insecure deserialization) 항목이 병합됐습니다. 소

프트웨어 및 데이터 무결성 오류는 애플리케이션이 신뢰할 수 없는 소스, 저장소 및 CDN, 플러그인, 라이브러리, 모듈에 의존하는 경우에 발생합니다. 안전하지 않은 CI/CD 파이프라인은 개발 및 배포 과정에서 애플리케이션이 변조되면 무결성이 훼손될 가능성이 있으므로, 애플리케이션이 사용하는 코드에 대한 무결성 검증 절차를 추가해야 합니다.

9. Security Logging and Monitoring Failures

기존에는 불충분한 로깅 및 모니터링(Insufficient Logging & Monitoring) 항목이었으나, 이번 개정에서 보안 로깅 및 모니터링 실패(Security Logging and Monitoring Failures) 항목으로 변경됐습니다. 적절한 로깅과 모니터링이 없다면 공격을 감지하고 대응할 수가 없기 때문에, 취약점 공격 예방뿐 아니라 공격 발생 감지 및 대응까지 포함하는 것으로 개정됐습니다.

10. Server-side Request Forgery

서버 측 요청 변조(Server-Side Request Forgery, SSRF)는 2021년에 신설된 항목입니다. 애플리케이션이 사용자 제공 데이터를 적절한 검증 없이 로컬 및 원격 리소스를 가져와 취약점을 발생시키는 상황을 의미합니다. 공격자는 SSRF 취약점이 존재한는 애플리케이션에서 서버 권한과 신뢰 관계를 이용해, 서버가 조작된 요청을 수행하도록 강제할 수 있습니다.